μISPs: Providing Convenient and Low-Cost High-Bandwidth Internet Access

José Brustoloni and Juan Garay

Information Sciences Research Center
Bell Laboratories, Lucent Technologies
600 Mountain Avenue, Murray Hill, NJ 07974, USA
`{jcb,garay}@research.bell-labs.com`

Abstract

We present μISP, a novel architecture for Internet Service Providers suitable for installation in airports, hotels, conference centers, cafés, and office or apartment buildings. Users access a μISP via a low-cost, high-bandwidth LAN, e.g. Ethernet or WaveLAN. A router connects the μISP's LAN to a shared high-bandwidth access link (e.g., DSL or cable) to a conventional ISP. For this service, a μISP charges its clients. The architecture supports a variety of payment methods, both offline (e.g., cash, credit card, or billing to a hotel room account) and online (e.g., eCash, SET, IBM Micro Payments, or Millicent). μISPs use IPSec's IKE protocol for securely exchanging authentication keys with paying users. Paying users use IPSec's AH protocol in tunnel mode to authenticate each packet they send. Therefore, μISPs can easily detect and drop packets of nonpaying users. A μISP must present to users a certificate signed by a recognized authority, but a user may simply present a self-signed certificate, as long as the user pays for service. Regardless of how online payment is implemented, it runs on the user's authenticated tunnel, and therefore can be securely bound to it. The μISP protocol allows users to monitor and control usage and supports recovery in case of a μISP or user computer crash.

Keywords: *Mobile access, access devices, electronic commerce, security and privacy.*

1   Introduction

Users typically connect to the Internet via Internet Service Providers (ISPs). In the conventional ISP architecture, illustrated in Figure 1, each user pays for the installation and maintenance of an access link connecting the user's computer(s) to an ISP's POP (Point of Presence). Alternatives for access links include dial-up PSTN (Public Switched Telephone Network) or ISDN (Integrated Services Digital Network) telephone lines, dedicated telephone lines (e.g., T1), DSL (Digital Sub-
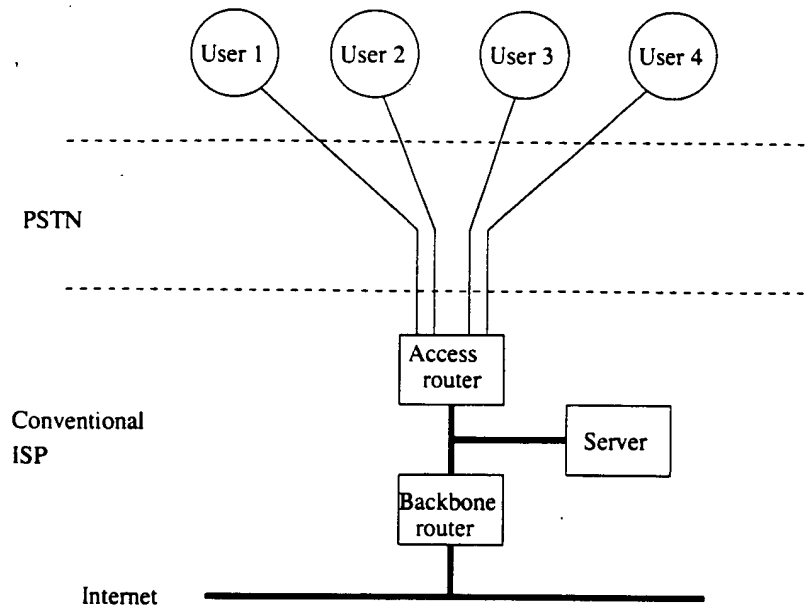
Figure 1: Dial-up PSTN lines are still the most common ISP access links. They provide low bandwidth and usually require one access link and ISP account per computer. When the user is mobile, dial-up PSTN lines may be unavailable, inconvenient, or expensive to use.

scriber Line), and cable. A contract between user and ISP lasts typically at least a month and often much longer than that.

The conventional ISP architecture has two shortcomings. First, access link costs are significant, i.e., correspond to a considerable fraction of the user's total Internet connection cost. Second, mobility support is poor, especially for higher access bandwidths. On the one hand, dedicated telephone lines, DSL, and cable can provide high bandwidths, but are available only where the user installs them (e.g, office or home). On the other hand, dial-up lines allow access wherever there is a phone, but phones may be unavailable, inconvenient, or expensive to use and, in any case, provide low bandwidth. For example, the number and location of pay-phones in airports, conference centers, and cafés is often adequate for short conversations but inappropriate for laptop hookups and Web browsing. Wireless phones are convenient in such situations, but are expensive to use for more than a few minutes. Additionally, long-distance calls may be necessary to dial-up the nearest POP of the user's ISP.

This paper introduces a novel architecture, μISP, that overcomes the above-mentioned shortcomings of conventional ISPs. A μISP reduces access costs by amortizing among many users the cost of a high-bandwidth access link to a conventional ISP. Users access a μISP via a low-cost, high-bandwidth Local Area Network (LAN), e.g., Ethernet or WaveLAN, as shown in Figure 2. The bandwidth dynamically allocated to each user is likely to be similar to that which many users enjoy at work, and much better than that afforded at home by a dial-up PSTN line. A router connects the μISP's LAN to the shared high-bandwidth access link, which may be, e.g., DSL or cable. For this service, the μISP charges its clients. The architecture supports a variety of offline
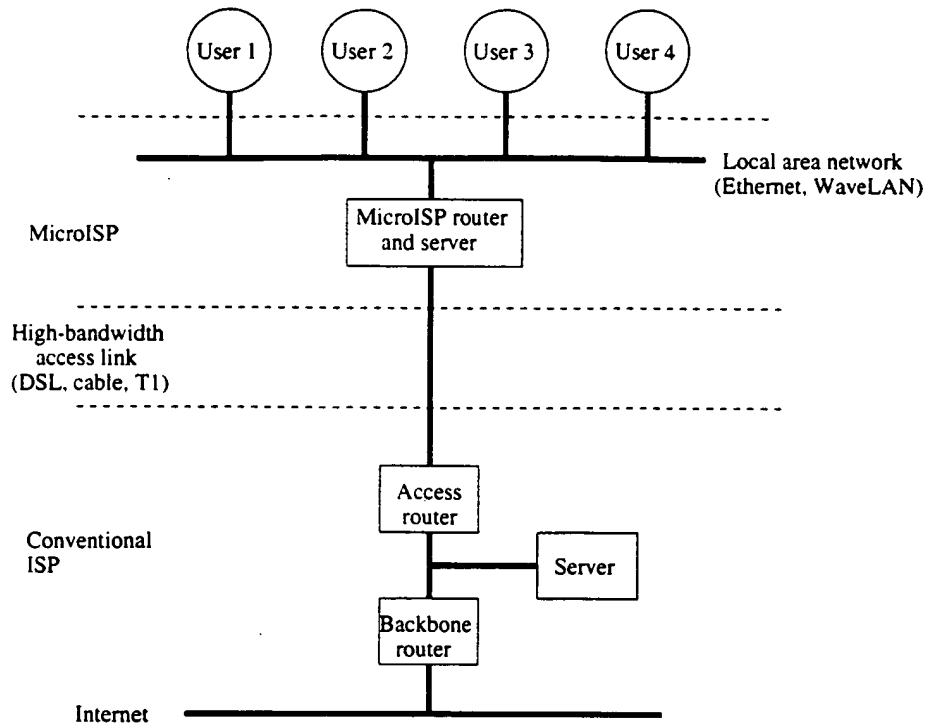
2

Figure 2: A μISP amortizes among many users the cost of a high-bandwidth access link (e.g., DSL). μISPs require little investment and space and therefore may be deployed widely and in convenient locations. μISPs support mobile users by allowing short-term contracts.

and online payment methods. μISPs support the needs of transient (mobile) users because, unlike conventional ISP contracts, μISP contracts can be short-term (e.g., may last only 15 minutes). The low investment necessary to set up a μISP and the potential profitability encourage widespread deployment.

A challenge in this type of architecture is how to prevent a nonpaying user from gaining free service or charging his or her Internet usage to a paying user. For example, if the μISP's LAN uses a shared medium, e.g. Ethernet or WaveLAN, a nonpaying user n might be able to communicate with another host h by (1) snooping in the LAN to find the IP address of an active (paying) user p, (2) sending packets to h by spoofing them, i.e., making the source address equal to p, and (3) receiving packets from h by snooping in the LAN to find packets with source equal to h and destination equal to p. The μISP architecture uses IPSec's standard Authentication Header (AH) in tunnel mode [18] to solve this problem. μISPs and paying users securely exchange authentication keys using IPSec's IKE (Internet Key Exchange) protocol [15, 24]. Paying users then use AH with such keys to authenticate each packet they send. No application (e.g., browser) modifications are necessary because IPSec is configured and operates at the network layer [19, 7]. Because AH authentication includes the packet's source address, and nonpaying users do not have authentication keys, the μISP can easily detect and drop spoofed packets, shutting off nonpaying users.

The μISP architecture uses IKE with certificate-based authentication. A μISP must present

to users a certificate signed by a recognized μISP Certifying Authority (CA). Users, on the other hand, need not incur the cost or trouble of obtaining certificates from a recognized CA. A user may present to μISPs a self-signed certificate and may thus remain anonymous if the payment method also preserves anonymity (e.g., cash or eCash [9]).

A potential source of complexity and implementation difficulty is the need to accommodate within the architecture the many different usage metrics and payment methods that may be desirable in a given μISP. Usage metrics may be, e.g., elapsed or usage time, or number of bytes or packets transmitted. Payment methods may be offline (e.g., cash or credit card) or online (e.g., eCash [9], SET [29], IBM Micro Payments [16], or Millicent [12]). The μISP architecture defines a carefully designed protocol that supports these and other options. In particular, the μISP protocol does not require modifications in online payment method implementations, because the protocol automatically and securely binds online payment (however implemented) with the above-mentioned AH tunnel.

Another potential pitfall is how to recover from computer crashes. In many scenarios, crashes are actually expected. For example, a guest at a hotel or conference center may turn her laptop on and off several times until her contract with the local μISP expires. The μISP protocol allows graceful recovery by issuing the user a *receipt* after the user pays. The μISP authenticates the receipt using a secret key. If the usage metric is simply elapsed time (flat fee until an expiration time), the receipt contains all the information necessary for recovery and, except for online payments, the μISP need not commit user state to stable storage. Recovery of crashes between the time the user sends online payment to the μISP and the time the user commits the respective receipt to stable storage is handled according to the respective payment method.

As mentioned above, μISPs use the services of conventional ISPs. μISPs may be able to reduce substantially the cost of such services by implementing NAPT (Network Address Port Translation) [30] in the router between the μISP LAN and the shared high-bandwidth access link. In this case, all μISP users use the same global IP address and appear to the conventional ISP as a single host.

Cybercafés are a competing alternative to ISPs. Typically, cybercafés lease to users desktop computers connected to the Internet. Like μISPs, cybercafés allow short-term service contracts. Cybercafés require much greater investment than do μISPs because cybercafés own the computers used and need space for them, whereas μISPs simply provide Internet access to user-owned laptop or desktop computers and therefore need negligible space. Consequently, μISPs may be deployed more widely and in more convenient locations. An additional advantage of μISPs is that users may find their own computers more familiar and secure to use than are those provided by a cybercafé. A hybrid service model, μISP café, is also possible, and would accommodate the latter users as well as users who do not have their computers with them.

Unlike conventional ISPs, cybercafés and μISPs do not themselves provide to users local content and email or Web page hosting services. However, this is becoming hardly a disadvantage, because users can easily find on the Web portals or servers that provide such services for free (e.g., www.yahoo.com, hotmail.com, and www.geocities.com). Web-based services have the advantage of being accessible wherever the user may be.

The rest of this paper is organized as follows. Section 2 gives an overview of the μISP pro-

tocol and its phases. Sections 3 to 8 describe each of the phases in greater detail: networking configuration, secure tunnel establishment, control channel establishment, contract establishment and binding, usage metering, and settlement. Section 9 discusses implications of using NAPT, and Section 10 discusses some other variations in the μISP design. Finally, Section 11 reports the performance of a prototype implementation, and Section 12 concludes.

2 μISP protocol overview

As shown in Figure 2, the main components of the μISP architecture are a high-bandwidth LAN, a router, a server, a shared high-bandwidth access link, and a protocol for communication between user computers and the μISP server and router. Although server and router can in general be implemented separately, in this paper we assume that the server is implemented within or as an adjunct to the router, and the term "μISP" is used to refer to that combination when communicating with users.

This section gives an overview of the μISP protocol. Its design was influenced by the following constraints:

- *Use of standard hardware and protocols.* μISP access will typically be based on a shared medium, and it is tempting to design special network cards, protocols, tokens, or smart-cards to guarantee secure access and payment in such networks. For example, cable access networks are also based on a shared medium and use special, certified cable modems and a special protocol (BPI+ [4]) to secure communications between cable modems and cable head-ends. (An offline process binds a certified cable modem with a user account, to which Internet usage is billed.) However, special hardware makes installation costly, and special protocols may not be as well scrutinized and secure as standard ones. For example, the protocol initially used in cable systems did not authenticate the cable head-end, leaving an important security hole. In contrast, μISPs interoperate with standard LAN cards that many users already own (e.g., Ethernet or WaveLAN), and use IETF's standard IPSec protocols [19]. IPSec is supported by most current operating systems, including Windows 2000, NT 5, and Linux.

- *Use of existing online payment method implementations.* There are many proposals for online payment methods (e.g., eCash [9], SET [29], CyberCash [3], IBM Micro Payments [16], or Millicent [12]). Some proposals are proprietary and no proposal has achieved wide use. Embedding specific online payment methods into μISP implementations would therefore involve considerable licensing and maintenance difficulties. These difficulties are avoided by μISP's secure binding of payments received by independently designed and implemented processes.

- *Support for offline payment.* Depending solely on online payment could adversely delay μISP deployment, given that adoption of online payment methods has been slow. Additionally, offline payment would be quite natural in many scenarios, e.g. when a user checks in at an airport, hotel, or conference, or comes into a lounge or café.

5

Consequently, the μISP protocol actually combines several underlying protocols and processes, some of which may be offline.

The μISP protocol defines the following phases:

1. *Networking configuration.* Before a user's computer can communicate with the μISP, some of its networking parameters need to be configured, e.g. the IP addresses of the user's computer and of the default router. μISP uses the standard Dynamic Host Configuration Protocol (DHCP) [8] to achieve this configuration.

2. *Secure tunnel establishment.* μISP uses IPSec's AH protocol in tunnel mode to authenticate user packets. The tunnel guarantees that all packets received by the μISP from a certain IP address correspond to the same user. The identity of that user is immaterial to the μISP, provided that the user pays for the μISP services. Therefore, the user may present a self-signed certificate. On the other hand, before paying, users may want to verify that they are communicating with a bona fide μISP. Therefore, μISPs must present a certificate signed by a recognized μISP certifying authority. Users are also given the options of having the secure tunnel (1) use IPSec's AH protocol to authenticate all packets in transit to the user, and (2) use IPSec's Encapsulating Security Payload (ESP) protocol [20] to encrypt all packets in transit between μISP and user.

3. *Control channel establishment.* The μISP needs a secure control channel to send to the user the μISP's price list before payment and a receipt after payment. The user also uses this channel to control his or her Internet usage. If the tunnel established in the previous phase uses ESP, the control channel is simply a TCP connection; otherwise, the control channel uses the TLS protocol [5].

4. *Contract establishment and binding.* In this phase, (1) the μISP presents to the user a list of options for service (e.g., usage metrics: elapsed or usage time, or number of bytes or packets transmitted) and payment (e.g., eCash, SET, IBM Micro Payment, or Millicent) and the respective prices, (2) the user selects the desired options, (3) the user makes a deposit payment, and (4) the μISP gives a receipt to the user. This phase is skipped entirely if the user's computer already has the receipt of an outstanding contract and the user, e.g., is reconnecting to the μISP again after turning his or her computer off. Steps 1 to 3 are skipped if the user presents a valid password, received from the μISP in offline processing of those steps (e.g., payment by cash, credit card, or billing to a hotel room account). Step 3 may be performed over channels other than the one established in the previous phase (e.g., in the case of independent online payment method implementation). However, any online payment will necessarily use the tunnel established in phase 2, and therefore can be securely bound to it.

5. *Usage metering.* Until a user's Internet usage reaches the amount selected in the previous phase, the user can send or receive packets to the Internet using the μISP's facilities. To monitor and control his or her usage, the user may exchange messages with the μISP, using

6

the control channel established in phase 3. These messages may, for example, suspend, resume, or terminate service.

6. *Settlement.* When service to a user terminates, the net amount paid by the user should be equal to his or her actual usage. If the deposit of phase 4 is greater than the net amount, the user may be due a refund. This settlement is performed in this final phase.

The following sections describe each phase in greater detail.

3   Networking configuration

This section describes in greater detail phase 1 of the μISP protocol.

In this phase, μISPs use DHCP for configuring the networking parameters of dynamically connected user computers. When user computers boot or restart, they broadcast in the LAN DHCP packets requesting configuration. A μISP server replies with the necessary parameters, including network mask and broadcast address, IP addresses of the user's computer and of the default router, and possibly the IP addresses of a DNS (Domain Name System) server, NTP (Network Time Protocol) server, and line printer server. μISPs use DHCP's dynamic IP address allocation, so that IP addresses assigned to a user's computer remain valid only during a specified lease time. User computers must periodically renew their leases to preserve their IP addresses. Expired IP addresses may be reused. In μISPs, the list of unallocated addresses is maintained in FIFO order.

DHCP makes μISPs easy to use: A user may, for example, link her laptop to the Ethernet or WaveLAN in an airport lounge or conference room, reboot the computer, and automatically be configured to access the Internet. DHCP is supported by most current operating systems, including Windows 2000, NT, and Linux.

4   Secure tunnel establishment

This section describes phase 2 of the μISP protocol and the use of IPSec [19] in μISPs.

IPSec defines two protocols for secure data communication, AH [18] and ESP [20]. These protocols are implemented at the network layer and therefore do not require modifications in user applications. AH can provide authentication of packet origin, proof of integrity of packet data, and protection against packet replay. ESP can provide, in addition to AH's services, encryption of packet data and limited traffic flow confidentiality. However, unlike AH's authentication, ESP's does not include the packet's source and destination IP addresses. AH and ESP can be used in either transport or tunnel mode, as illustrated in Figure 3. Transport mode provides end-to-end security between the packet's source and destination. In contrast, tunnel mode encapsulates packets and thus provides security between the nodes where the packet is encapsulated and decapsulated.

μISPs use the AH protocol in tunnel mode to authenticate all packets received from a user's IP address, guaranteeing that the respective address is always used by the same user while the address is allocated or bound to a contract. If the user so selects, the μISP may also use AH in tunnel mode to authenticate all packets sent or forwarded to the user. In this case, after the tunnel is established,
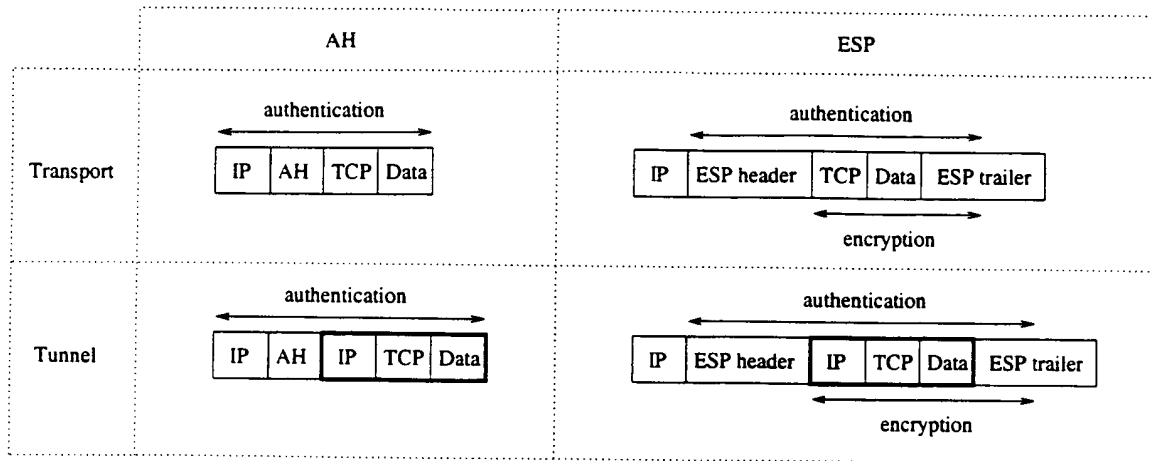
Figure 3: IPSec packet format depends on protocol (AH or ESP) and mode (transport or tunnel). The portion of the packet that is authenticated or encrypted is different for AH or ESP. The encapsulated packet is shown in bold.

the user may verify the networking configuration performed insecurely by DHCP in the previous phase. The authenticated tunnel and the verification of the networking configuration can prevent, e.g., DHCP- or DNS-based security attacks against the user on the μISP's LAN. Another option available to users is to use ESP encryption for all packets sent to or received from the user's address. This option preserves privacy on the μISP's LAN. The authentication and encryption options are most useful when users are accessing insecure sites on the Internet. A user may not need these options, e.g., when he or she establishes another IPSec tunnel *within* the user's μISP tunnel to communicate securely with an IPSec gateway into the Intranet of the user's employer. In the latter case, the nested tunnel may already provide all necessary authentication and encryption.

Another IPSec protocol, IKE [15], establishes security associations that define the algorithms and cryptographic keys used by AH and ESP. Security associations have a specified lifetime, after which they are terminated and need to be replaced [19]. The μISP protocol uses IKE authenticated with signatures. The initiator is always the user. Using this method, μISP and user perform a Diffie-Hellman [6] key exchange for securely establishing a shared secret, from which AH and ESP keys are derived. Each party then authenticates the other by verifying the other's signature [26] on a message containing the other's certificate. A party's certificate contains that party's public key, which is necessary for verifying that party's signatures. A party's certificate also contains that party's identity and is itself usually signed by a certifying authority (CA) whose public key is widely known, so that any party can verify the certificate. Certificate formats are defined, e.g., in the X.509 standard [33]. Authentication is necessary to prevent "person-in-the-middle" attacks, where an intruder would pretend to be the user when communicating with the μISP and to be the μISP when communicating with the user. Therefore, μISPs must present certificates signed by a recognized μISP CA, which maintains registration procedures appropriate for such certification. In a PKIX-based implementation [17], these certificates would contain a policies extension with explicit-text user notice. This notice should be displayed to the user [17] and informs the location

8

and type of LANs supported by the μISP. On the other hand, the μISP does not really need to authenticate the user's identity in this phase; the μISP's only requirement is that the user pay in phase 4 of the protocol, and that no other user be able to use that payment to gain service. Therefore, the μISP can be configured to accept self-signed user certificates in IKE exchanges. Using such certificates, users can remain anonymous.

IPSec security policies are defined in a Security Policy Database (SPD) per network interface [19]. Each SPD entry specifies a selector and a rule. Selectors may match, e.g., packets that have a certain protocol and source and destination IP addresses and port numbers (ranges and wild cards are allowed for these values). Actions may be to drop the packet, bypass IPSec, or apply specified IPSec protocols to the packet. The SPD of the LAN interface of μISP routers is configured, in the incoming case, to bypass IPSec in the cases of DHCP and IKE packets destined to the μISP, to perform AH and optionally ESP processing to packets whose source address is bound (phase 4) to an active contract or whose destination is the μISP, and to drop remaining packets. In the outgoing case, the SPD is configured to bypass IPSec in the cases of DHCP and IKE packets whose source is the μISP, to bypass IPSec or apply AH or ESP processing to packets whose source is the μISP or whose destination address is bound to an active contract, and to drop remaining packets. While a user's computer is accessing a μISP, the SPD of the computer's LAN interface is similarly configured, with the incoming and outgoing cases reversed.

5   Control channel establishment

This section briefly covers phase 3 of the μISP protocol.

Because of its uses in the μISP protocol, the control channel should guarantee message authenticity and privacy in both directions. Privacy is needed, e.g., to prevent the eavesdropping of receipts and their later use by nonpaying users. If the user selected the privacy option in phase 2, all communication over the user's tunnel is already secured in both directions by ESP. Therefore, the user establishes the control channel by simply opening a TCP connection to a well-known port in the μISP. Otherwise, the tunnel established in phase 2 does not provide all the required security (i.e., only authenticates user packets to the μISP). Therefore, the user employs the TLS [5] protocol for establishing a secure control channel over the user's tunnel. The principals of the TLS channel are guaranteed to be the same as those of the AH tunnel: The μISP is authenticated using its certificate, while the user is authenticated by AH.

6   Contract establishment and binding

This section describes how a contract between μISP and user is established in offline and online cases and how the IP address assigned to the user in phase 1 and secured in phase 2 is bound to the user's contract in phase 4 of the μISP protocol.[1]

The contract is established in four steps:

---

[1]Frameworks for online price negotiation and payment selection have been discussed in the e-commerce literature and there are proposals for their standardization (e.g., JEPI [32] and SEMPER [28]).

1. *μISP offer.* The μISP presents to the user a contract form containing a serial number, the current date and time, available service options, including acceptable usage metrics (e.g., elapsed or usage time, or number of bytes or packets transmitted) and the respective prices, and acceptable payment methods (e.g., in offline cases: cash, credit card, or billing to an account, such as a hotel room account; or in online cases: *e*Cash [9], SET [29], IBM Micro Payments [16], or Millicent [12]). A contract is always subject to an expiration time. Prices may depend, e.g., on whether the user has selected phase 2's privacy option, on the amount of usage, on the payment method selected, and on the current or anticipated μISP load.

2. *User request.* The user completes the form indicating the desired usage metrics, soft and hard usage limits, and payment method. In offline cases, if the payment method is not cash, the user physically signs the form.

3. *User deposit.* The user employs the selected payment method to deposit with the μISP an amount equal to the selected hard usage limit. If the payment method is credit card or SET, this deposit is implemented by an authorization transaction. In certain online cases (which do not include SET or *e*Cash), the μISP may need to allow the user to communicate directly with external servers before paying. In IBM Micro Payments [16], for example, the user may need to contact his or her issuer to obtain the user's daily certificate, which is necessary for making payments. As another example, in Millicent [12], the user may need to contact his or her broker to convert broker scrips into μISP scrips (scrips are Millicent's merchant-issued payment instruments). To enable the latter payment methods, modifications in IPSec's SPDs may be necessary (e.g., permitting user communication with certain supported issuers or brokers for a limited time).

4. *μISP receipt.* The μISP gives to the user a copy of the contract and password (offline cases) or a receipt (online cases). The user commits the receipt to stable storage. The receipt is a data structure that includes the contract's serial number, date and time, expiration, selected usage metrics and limits, and payment parameters. The μISP authenticates the receipt with a Message Authentication Code (MAC). MAC computation uses a secret key with, e.g., the DES-MAC [27], keyed-MD5 [31], or HMAC [1] algorithms.

Phase 4 of the μISP protocol is executed as follows. If the user's stable storage contains the receipt of an outstanding contract, the user sends the receipt over the control channel to the μISP, which verifies that the contract is still outstanding, is not bound to an IP address, and is not being settled. The μISP then binds the contract with the user's IP address, concluding this phase. Otherwise, if the user sends over the control channel the password of an unbound outstanding offline contract, the μISP binds the contract with the user's IP address and returns the corresponding receipt. The user then commits the receipt to stable storage, concluding this phase. Otherwise, the user sends over the control channel a request for online contract establishment, triggering the four steps described above. The contract is bound to the user's IP address in step 4.
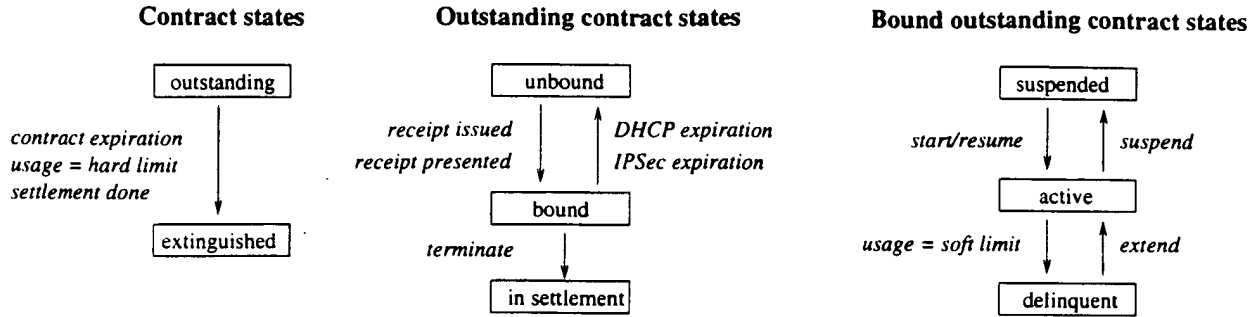
Contract states   **Outstanding contract states**   **Bound outstanding contract states**

```
    ┌─────────────┐              ┌──────────┐                    ┌─────────────┐
    │ outstanding │              │ unbound  │                    │  suspended  │
    └─────────────┘              └──────────┘                    └─────────────┘
```

contract expiration   receipt issued   DHCP expiration   start/resume   suspend
usage = hard limit   receipt presented   IPSec expiration
settlement done

```
    ┌──────────────┐              ┌──────────┐                    ┌─────────────┐
    │ extinguished │              │  bound   │                    │   active    │
    └──────────────┘              └──────────┘                    └─────────────┘
```

                            terminate   usage = soft limit   extend

```
                              ┌──────────────┐                    ┌─────────────┐
                              │ in settlement│                    │  delinquent │
                              └──────────────┘                    └─────────────┘
```

Figure 4: Contract states and transitions. The initial states are the top ones.

7 Usage metering

This section describes phase 5 of the μISP protocol, usage metering. Metering depends on contract and IP address states. Therefore, this section discusses the possible states and the events that trigger state transitions (e.g., user commands).

Metering is greatly simplified when the usage metric is elapsed time. In such cases, the μISP does not need to commit contract state and usage to stable storage in order to be able to recover from μISP crashes: The receipts themselves contain all the necessary information. If the μISP's shared link to a conventional ISP is charged on a similar basis, this is probably the best alternative. However, if the μISP is charged according to number of bytes or packets transmitted, it may need to do the same with respect to its clients.

In general, users monitor and control contract state by sending commands to the μISP. Available commands include report usage, start/resume, suspend, or terminate a contract, and extend a contract's soft limit up to its hard limit. The μISP replies to these commands include the accumulated usage and the soft and hard limits. The μISP may also send asynchronous warning messages when a contract's accumulated usage reaches its soft or hard limit or expires, or when a user on a different IP address presents the contract's receipt on phase 4 of the μISP protocol. All these messages contain the contract's serial number and are sent over the control channel.

A contract is *outstanding* between its establishment and settlement, and becomes *extinguished* when it expires, has accumulated usage that reaches its hard limit, or after settlement, as shown in Figure 4. An outstanding contract can be *unbound*, *bound*, or *in settlement*. An outstanding contract is initially unbound. An unbound contract becomes bound to an IP address when a receipt for that contract is sent to or from that IP address in phase 4 of the μISP protocol. A bound contract becomes unbound when the user's computer lets the respective DHCP lease or IPSec security association expire (e.g., because of a crash). An unbound or bound contract becomes in settlement when the user issues a terminate command.

A bound contract can be *suspended*, *active*, or *delinquent*. A bound contract is initially suspended. A suspended contract becomes active when the user issues a start/resume command. An active contract becomes suspended when the user issues a suspend command, and becomes delinquent when the usage reaches the soft limit. A delinquent contract becomes active when the user issues a valid extend command.
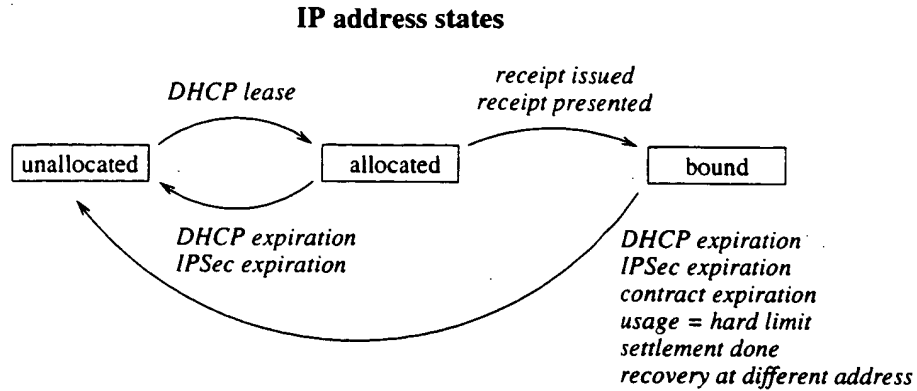
11

IP address states

Figure 5: IP address states and transitions. The initial state is *unallocated*.

An IP address can be *unallocated, allocated,* or *bound* to a contract, as shown in Figure 5. IP addresses are initially unallocated. An unallocated IP address becomes allocated when DHCP allocates it to a user's computer. An allocated IP address becomes unallocated again if the user's computer allows the respective DHCP lease or IPSec security association to expire. An allocated IP address becomes bound to a contract when the converse is true. A bound IP address b becomes unallocated when the respective contract becomes unbound or extinguished or (1) a user on a different IP address d presents the contract's receipt on phase 4 of the μISP protocol; and (2) the μISP repeatedly warns b but b does not respond, which suggests that the user's computer crashed and is now recovering on a different address.

The μISP meters a contract's usage time only while the contract is active. The μISP router forwards to or from the Internet and meters the number of bytes or packets only of packets that use an IP address bound to an active contract. The μISP also allows packets whose source or destination is the μISP.

8   Settlement

This section describes phase 6 of the μISP protocol. This is the final phase.

If the user lets the contract expire or uses the contract fully to its hard limit, the μISP retains the whole deposit. If the payment method is credit card or SET, the μISP automatically performs a settlement transaction for that value. On the other hand, if the usage is below the hard limit, an adjustment or refund is necessary, and will be processed according to the payment method. In offline cases other than cash, the user physically signs a new form. In the credit card and SET cases, a settlement transaction for the value of the actual usage is performed. In the cash, *e*Cash [9], and Millicent [12] cases, a refund is returned to the user. In the cases of offline billing to an account and of IBM Micro Payments [16], the μISP simply adjusts its billing records.

9 NAPT (Network Address Port Translation)

μISPs have to allocate one IP address per contract that is bound or in settlement. In order to get more than one IP address from a conventional ISP, the μISP will typically need to pay extra. A cost-saving alternative is to have the μISP router implement NAPT (Network Address Port Translation) [30] so that all μISP users share a single global IP address. However, NAPT may not support certain protocols, as discussed in this section.

If a μISP implements NAPT, the IP addresses that it allocates to user computers are private (unregistered and possibly not globally unique) [25]. NAPT translates respectively the source or destination IP addresses and port numbers (between private and global) in packets sent to or received from the Internet. NAPT builds a translation table such that all local hosts share the same global IP address, and destinations of packets received from the Internet can be determined by the global port numbers.

NAPT is not transparent to application-layer protocols that include host addresses or port numbers within their payloads (e.g., FTP). For each such protocol, NAPT requires a corresponding Application Level Gateway (ALG) that knows how to modify the application-level payloads. ALG support has been increasing, but NAPT may still not interoperate with certain protocols [30].

NAPT is also not transparent with respect to most of IPSec's protocols and modes. (However, note that NAPT does not affect the AH tunnel used in the μISP protocol, which terminates at the router.) NAPT address translations invalidate AH's authentication, whose scope includes the source and destination addresses. NAPT address translations can also invalidate TCP or UDP checksums of ESP packets in transport mode, which are calculated including the packet's source and destination addresses. An ALG for these cases is not possible because it would need to know the cryptographic keys used, which would violate end-to-end security. NAPT can, however, interoperate with ESP in tunnel mode, where the entire packet is encapsulated and NAPT's translations do not affect the encapsulated checksum. VPN Masquerade [14] is a NAPT implementation that provides such support. VPN Masquerade uses a number of heuristics for demultiplexing incoming traffic. These heuristics are subject to race conditions and may, in some cases, fail, causing packet misdelivery. However, packet misdelivery does not break security any more than eavesdropping in the LAN would.

We recently proposed [2] a DHCP extension that allows hosts to lease from NAPT, e.g., global IP addresses and port numbers. Using that extension, IPSec implementations can always interoperate correctly with NAPT. Another alternative currently being considered is to replace NAPT by RSIP (Realm-Specific IP) [30]. We believe that either solution will go a long way toward eliminating NAPT's current interoperation difficulties. Therefore, the cost-reducing option of using a single global IP address in μISPs may soon become even more attractive.

10 Other design variations

Many details is the μISP design can be altered without essentially impacting in the overall functionality. This section discusses some of the possible modifications.

13

An obvious variation is to use another protocol for authentication and/or encryption in the secure tunnel. The new protocol must be able to encapsulate and decapsulate packets. For example, instead of AH, a μISP might employ ESP's authentication option to authenticate packets sent by paying users. In either case, ESP's encryption is optional, and tunnel mode is used. Unlike AH, ESP's authentication does not cover the packet's source and destination IP addresses. However, ESP's authentication does cover the entire encapsulated packet. Therefore, ESP's authentication is sufficient for spoofing prevention [13].

Another variation would be to set up the control channel *before* the secure tunnel. The control channel might allow, for example, the transmission of cryptographic keys to be used in the tunnel. In this case, IKE authentication could, e.g., use a pre-shared key, instead of digital signatures.

Other variations include using a solution other than the DHCP protocol for configuring networking parameters of user computers; using a solution other than the IKE protocol for establishing the secure tunnel's cryptographic algorithms and keys; using a firewall, instead of IPSec's SPDs, for dropping packets of nonpaying users; or using a protocol other than TLS (e.g., SSL [11]) for the secure control channel.

11 Performance

An important practical question is how powerful a computer would be necessary for implementing a μISP router/server. To answer this question, we built a PC-based μISP router/server prototype and report its performance in this section.

Our prototype is based on a low-cost PC with a 400 MHz Pentium II CPU and 64 MB of main memory. The prototype uses the freely available Linux 2.2.12 operating system and the FreeS/WAN 1.1 IPSec implementation [10].

To circumvent limitations of our prototype environment, we used several of the design alternatives discussed in the previous section. First, we used SSL instead of TLS because we had an SSL implementation easily available. Second, the prototype uses SSL to establish the control channel *before* the secure tunnel, because FreeS/WAN 1.1 does not fully implement IKE. The control channel securely transmits randomly generated keys for FreeS/WAN authentication using pre-shared keys.

We connected a client and a server Linux PCs to the μISP prototype using separate 10 Mbps Ethernets, and measured the TCP throughput between the client and server. For control, we also measured the TCP throughput between client and server when connected on the same 10 Mbps Ethernet (without the μISP): 6.4 Mbps. When client and server were connected through the prototype implementing only routing and NAPT (no μISP functionality), the TCP throughput dropped slightly to 6.2 Mbps, and the CPU utilization on the prototype was 4%. With μISP functionality and packet authentication between client and μISP (using AH with MD5 [22]),[2] the TCP throughput between client and server was 5.8 Mbps and the CPU utilization on the prototype was 26%. Finally, with μISP functionality and both authentication and encryption between client and μISP

---

[2] For ease of implementation, AH was used in both directions. Results would probably improve significantly if AH were used only from client to μISP, as described in Section 4.

(using ESP with MD5 and triple DES [21]), the TCP throughput between client and server was 5.3 Mbps, and the CPU utilization on the prototype was 70%.

We also measured the time necessary for a client to connect to the μISP prototype (steps 1 to 4 of the μISP protocol), as well as the load imposed on the prototype's CPU by such connections (with no other network or CPU activity). We simultaneously started connections from two 100 MHz Pentium clients and one 700 MHz dual-processor Pentium III client. Connection took 0.5 s for the fast client and 1.9 and 2.1 s for the slow clients. The prototype CPU was 31% utilized during these connections.

These measurements suggest that even a modest PC can handle the loads that may be expected on a μISP router/server. Access links such as T1, DSL and cable provide bandwidths from 0.6 to 7 Mbps downstream and from 0.6 to 1.5 Mbps upstream (cable can theoretically support up to 27 Mbps downstream, but cable modems usually limit a client's bandwidth to 1 Mbps). Such bandwidths are one to two orders of magnitude greater than those enabled by PSTN (57 Kbps downstream and 33 Kbps upstream), but still represent only a moderate load for today's processors.

The measurements also justify charging a premium price for privacy on the μISP's LAN: ESP's authentication (MD5) and encryption (triple DES) imposed a much higher load on the μISP prototype than did AH's authentication (MD5) alone.

12 Summary

We described μISP, a novel architecture that allows Internet access services to be securely provided and charged over standard shared-medium LANs, e.g. Ethernet or WaveLAN. Such LANs can be easily installed in convenient locations, e.g. airports, hotels, conference centers, lounges, and cafés. Standard protocols automatically configure user computers so that they can access the Internet. A router connects the μISP LAN to a shared high-bandwidth access link (e.g., DSL or cable) to a conventional ISP. Because a μISP amortizes the cost of the access link among many users, it can also reduce the cost of Internet access in offices and apartment buildings. The bandwidth dynamically allocated to each user is likely to be similar to that enjoyed by many users at work, and much better than that provided by a dial-up PSTN line. The architecture supports a variety of payment methods, both offline (e.g., cash, credit card, or billing to a hotel room account) and online (e.g., eCash, SET, IBM Micro Payments, or Millicent). μISPs use IPSec's IKE protocol for securely exchanging authentication keys with paying users. Paying users use IPSec's AH protocol in tunnel mode to authenticate each packet they send. Therefore, μISPs can easily detect and drop packets of nonpaying users. A μISP must present to users a certificate signed by a recognized μISP certifying authority. However, μISPs can accept users who do not have a certificate by a recognized authority or who choose to remain anonymous, as long as those users provide valid payment. The μISP protocol can use independent online payment method implementations because, regardless of how online payment is implemented, it runs on the user's authenticated tunnel, and therefore can be securely bound to it. The μISP protocol allows users to monitor and control usage and supports recovery in case of a μISP or user computer crash.

References

[1] M. Bellare, R. Canetti, and H. Krawczyk, "Keyed Hash Functions and Message Authentication," in *Advances in Cryptology–Crypto '96*, N. Koblitz, ed., Lecture Notes in Computer Science No. 1109, Springer-Verlag, 1996, pp. 1-15.

[2] J. Brustoloni and J. Garay. "Application-Independent End-to-End Security in Shared-Link Access Networks," in *Proc. IFIP Networking'2000*, Lecture Notes in Computer Science, Springer-Verlag, May 2000.

[3] CyberCash. Home page at http://www.CyberCash.com/.

[4] Data-Over-Cable Service Interface Specifications, "Baseline Privacy Plus Interface Specification," SP-BPI+-101-990316, CableLabs, 1999. Available at http://www.-cablemodem.com.

[5] T. Dierks and C. Allen. "The TLS Protocol Version 1.0," IETF, RFC 2246, Jan. 1999.

[6] W. Diffie and M.E. Hellman. "New directions in cryptography," in *Transactions on Information Theory*, IEEE, IT-22: 644-654, 1976.

[7] N. Doraswamy and D. Harkins. "IPSec: The New Security Standard for the Internet, Intranets and Virtual Private Networks," Prentice-Hall, 1st. ed., July 1999.

[8] R. Droms. "Dynamic Host Configuration Protocol," IETF, RFC 2131, Mar. 1997.

[9] *e*Cash Technologies, Inc. Home page at http://www.ecashtechnologies.com/.

[10] FreeS/WAN. Homepage at http://www.xs4all.nl/~freeswan/.

[11] A. Freier, P. Karlton and P. Kocher. "The SSL Protocol Version 3.0," Netscape, Mar. 1996. Available at http://home.netscape.com/eng/ssl3/ssl-toc.html.

[12] S. Glassman, M. Manasse, M. Abadi, P. Gauthier and P. Sobalvarro. "The Millicent Protocol for Inexpensive Electronic Commerce," in *Proc. 4th Intl. World Wide Web Conference*, W3C, Boston, MA, Dec. 1995. Available at http://www.w3.org/conferences/WWW4/.

[13] R. Glenn and S. Kent. "The NULL Encryption Algorithm and Its Use With IPsec." IETF, RFC 2410, Nov. 1998.

[14] J. Hardin. "Linux VPN Masquerade." Homepage at http://www.wolfenet.com/-~jhardin/ip_masq_vpn.html.

[15] D. Harkins and D. Carrel. "The Internet Key Exchange (IKE)," IETF, RFC 2409, Nov. 1998.

[16] A. Herzberg and H. Yochai. "MiniPay: Charging per Click on the Web," in *Proc. 6th Intl. World Wide Web Conference*, W3C, Santa Clara, CA, 1997. Available at http://www.scope.gmd.de/info/www6/. Now called IBM Micro Payments. Home page at http://www.hrl.il.ibm.com/mpay/.

[17] R. Housley, W. Ford, W. Polk and D. Solo. "Internet X.509 Public Key Infrastructure Certificate and CRL Profile," IETF, RFC 2459, Jan. 1999.

[18] S. Kent and R. Atkinson. "IP Authentication Header," IETF, RFC 2402, Nov. 1998.

[19] S. Kent and R. Atkinson. "Security Architecture for the Internet Protocol," IETF, RFC 2401, Nov. 1998.

[20] S. Kent and R. Atkinson. "IP Encapsulating Security Payload (ESP)," IETF, RFC 2406, Nov. 1998.

[21] C. Madson and N. Doraswamy. "The ESP DES-CBC Cipher Algorithm with Explicit IV," IETF, RFC 2405, Nov. 1998.

[22] C. Madson and R. Glenn. "The Use of HMAC-MD5-96 within ESP and AH," IETF, RFC 2403, Nov. 1998.

[23] C. Madson and R. Glenn. "The Use of HMAC-SHA-1-96 within ESP and AH," IETF, RFC 2404, Nov. 1998.

[24] D. Maughan, M. Schertler, M. Schneider and J. Turner. "Internet Security Association and Key Management Protocol (ISAKMP)," IETF, RFC 2408, Nov. 1998.

[25] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot and E. Lear. "Address Allocation for Private Internets," IETF, RFC 1918, Feb. 1996.

[26] R. Rivest, A. Shamir and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," in *Communications of the ACM*, ACM, Vol. 21(2), Feb. 1978.

[27] B. Schneier. "Applied Cryptography," 2nd. ed., John Wiley & Sons, New York, NY, 1996.

[28] Secure Electronic Marketplace for Europe (SEMPER). Homepage at http://www.semper.org/.

[29] SET. Homepage at http://www.setco.org/.

[30] P. Srisuresh and M. Holdrege. "IP Network Address Translator (NAT) Technology and Considerations," IETF, RFC 2663, Aug. 1999.

[31] G. Tsudik, "Message Authentication with One-Way Hash Functions," *Proc. INFOCOM '92*, IEEE, 1992, pp. 2055-2059.

[32] W3C Joint Electronic Payments Initiative (JEPI). Homepage at
http://www13.w3.org/ECommerce/Overview-JEPI.html.

[33] Recommendation X.509 (1197 E). "Information Technology - Open Systems Interconnection - The Directory: Authentication Framework," ITU-T.

Vitae

José C. Brustoloni received his Ph.D. in Computer Science from Carnegie Mellon University in 1997 and has been with Bell Labs since then, where he works in the Network Systems Research Department. His main research interests are in access networks, quality of service and billing, programmable networks, protocol performance, and embedded systems.

Juan A. Garay received his Ph.D. in Computer Science from Penn State University in 1989. Before joining the Secure Systems Research Department at Bell Labs, he was with IBM's T.J. Watson Research Center in Yorktown Heights, NY. His areas of interest include the design and analysis of cryptographic protocols, distributed computing, and fault tolerance.